The University of Warwick

THEORY OF COMPUTATION

REPORT NO. 67

MONOTONE BOOLEAN FUNCTIONS COMPUTABLE BY PLANAR CIRCUITS

Meurig Beynon

Department of Computer Science
University of Warwick
COVENTRY CV4 7AL
England.

September 1984

Monotone boolean functions computable by planar circuits

Meurig Beynon

Dept. of Computer Science, University of Warwick,
COVENTRY CV4 7AL, UK.

ABSTRACT

A criterion for testing whether a given monotone boolean function f is planar monotone computable from the sequence of inputs $x_1, x_2, \ldots, x_n$ is developed in conjunction with an algorithm which (in principle) can construct a planar monotone circuit for f whenever one exists. Both the algorithm and the criterion require precomputation of the prime implicants and clauses of f.

As an application of the theory, it is shown that monotone boolean functions whose prime implicants and clauses contain configurations of a particular type cannot be computed by planar monotone circuits. All monotone boolean functions on 4 (or fewer) inputs are shown to be planar monotone computable.

Introduction.

In a recent paper [4], W.F.McColl has shown that the class of monotone boolean functions of n arguments $x_1,x_2,\ldots,x_n$ which can be realised by planar monotone circuits from the left-right sequence of inputs $x_1,x_2,\ldots,x_n$ includes the threshold functions T_2^n for $n\geq 2$, but excludes T_3^n for $n\geq 5$. The remarkable nature of McColl's planar monotone circuit for T_2^n, and the sufficient conditions for functions to be non-planar computable which he derives prompt the question:

> when can a given monotone boolean function $f(x_1,x_2,\ldots,x_n)$ be realised by a planar circuit from the sequence of inputs $\langle x_i\rangle$?

In this paper, an algorithm for deciding this question is outlined. Since the algorithm depends upon precomputation of the prime clauses and prime implicants of f, it is practical only for small values of n, but in principle, on input f, it will either construct a planar monotone circuit for f, or show that planar monotone realisation of f is impossible. If a yes/no answer is all that is required, a simple criterion can be applied directly to the prime implicants and prime clauses.

As easy consequences of the results required to justify the algorithm, it can be seen that all monotone boolean functions with at most 4 arguments are realisable by a planar monotone circuit, and that (predictably) a boolean monotone function can be computable with respect to one ordering of inputs but not another.

Background definitions and notation.

For the formal definition of a planar monotone circuit, see [4]. A brief explanation of the principal notations used is given below; for further details, see [1] or [2]. The algebra of all monotone boolean functions of $x_1,x_2,\ldots,x_n$ (viz. the finite distributive lattice freely generated by $x_1,x_2,\ldots,x_n$) will be denoted by FDL(n). If $f\in FDL(n)$, then P_f will denote the set of prime implicants of f, and Q_f the set of prime clauses. For g in FDL(n), the notation $P_f[g]$ denotes $\{p\in P_f \mid p\leq g\}$ and $Q_f[g]$ denotes $\{q\in Q_f \mid q\geq g\}$. A necessary and sufficient condition for g to be replaceable by h when computing f via a monotone circuit (see [3], also [1] and [2]) is that $P_f[h]\supseteq P_f[g]$ and $Q_f[h]\supseteq Q_f[g]$; this relation is denoted by $h\sqsubset_f g$.

Planar circuits and fringes.

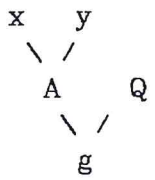
A sequence $y_1,y_2,\ldots,y_N$ of monotone boolean functions (denoted $\langle y_i\rangle$) will be called a ***fringe*** if the left-to-right sequence of functions $y_1,y_2,\ldots,y_N$ can be realised from the left-to-right sequence of inputs $x_1,x_2,\ldots,x_n$ by means of a planar monotone circuit.

Let f be a monotone boolean function of the inputs $x_1,x_2,\ldots,x_n$. The function f is planar monotone computable from the fringe $\langle y_i\rangle$ if there is a planar monotone circuit which computes f from the left-to-right sequence $y_1,y_2,\ldots,y_N$ of inputs; $\langle y_i\rangle$ is then an f-***fringe***. The function y_j in the f-fringe $\langle y_i\rangle$ is ***redundant*** (relative to the f-fringe $\langle y_i\rangle$) if

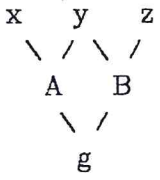
$$y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_N$$

is an f-fringe.

Suppose that f is planar monotone computable from $x_1, x_2, \ldots, x_n$, and that Γ is a planar monotone circuit for f. Provided that the value of f depends upon 3 or more inputs, there is a gate g in Γ at depth at least 2, and hence a gate at depth 2. Up to left-right symmetry, there are essentially two forms which the circuit defining the gate g can take:



Case 1.



Case 2.

In the diagrams, the symbols A and B denote single gates, and Q denotes an input to the gate g defining a non-constant function of at most two inputs z and t both distinct from y. In case 2, it may be assumed that A and B are gates of the same type, since g is otherwise a redundant gate. Assuming no redundancy in the input fringe, x,y in case 1, and x,y,z in case 2, will be consecutive inputs in the sequence $x_1, x_2, \ldots, x_n$.

In case 1, f is then planar monotone computable from the sequence

$$x_1, x_2, \ldots, x \equiv x_{j-1}, x \,\omega\, y, x_{j+1}, \ldots, x_n, \text{ where } y \equiv x_j, \text{ and } \omega \text{ is } \wedge \text{ or } \vee.$$

In case 2, f is planar monotone computable from

$$x_1, x_2, \ldots, x \equiv x_{j-1}, x \,\omega\, y, y \,\omega\, z, z \equiv x_{j+1}, \ldots, x_n, \text{ where } y \equiv x_j, \text{ and } \omega \text{ is } \wedge \text{ or } \vee.$$

Thus, if f is planar monotone computable from $x_1, x_2, \ldots, x_n$, there is a sequence of f-fringes

$$F_0 \equiv \langle x_i \rangle, F_1, \ldots, F_t \equiv \langle f \rangle$$

such that F_{k+1} is constructed from $F_k \equiv \langle y_i \rangle$ in one of three ways:

either by replacing y_j by $y_j' \equiv y_j \,\omega\, y_{j\pm1}$ - performing a "ω-diop" at y_j,

or by replacing y_j by $y_L \equiv y_j \,\omega\, y_{j-1}$, $y_R \equiv y_j \,\omega\, y_{j+1}$, - performing a "$\omega$-triop" at y_j,

or by removing a redundant function y_j from F_k.

In characterising planar monotone computable functions, it will be convenient to observe that a diop can be simulated by a triop followed by the elimination of a redundant gate. Thus (without loss of generality) every planar monotone circuit can be conceived as defined by a sequence of fringes, each of which is derived from its predecessor either by performing a triop or by deleting a redundant gate. A planar circuit for f constructed in this fashion will be called a ***fringed*** circuit for f.

The construction of fringed circuits.

In this section, it is shown that the operations on fringes required to construct a fringed circuit for f must be performed subject to appropriate pre-conditions. It will subsequently be seen that such operations on fringes preserve certain geometric features; this will provide a means of showing that a fringe is not an f-fringe.

The following theorem describes pre-conditions on $\wedge$-triops only; the dualisation simply entails the replacement of prime implicants by prime clauses:

Theorem 2.1.

Suppose that $\langle y_i \rangle$ is an f-fringe, and let $P_i \equiv P_f[y_i]$.

For $y_1, \ldots, y_{j-1}, y_L \equiv y_{j-1} \wedge y_j, y_R \equiv y_j \wedge y_{j+1}, y_{j+1}, \ldots, y_N$ to be an f-fringe, it is both necessary and sufficient that

$$P_j \subseteq P_{j-1} \cup P_{j+1}. \tag{2.1}$$

Proof:

Assume (2.1). Since $P_f[y_L \vee y_R] = P_f[y_j \wedge (y_{j-1} \vee y_{j+1})] = P_j \cap (P_{j-1} \cup P_{j+1}) = P_j$, and

$$Q_f[y_L \vee y_R] = Q_f[y_j \wedge (y_{j-1} \vee y_{j+1})] = Q_j \cup (Q_{j-1} \cap Q_{j+1}) \supseteq Q_j,$$

it follows that $y_L \vee y_R \sqsubset_f y_j$. Hence a planar circuit computing f from $\langle y_i \rangle$ can be adapted to compute f from

$$y_1, \ldots, y_{j-1}, y_L, y_R, y_{j+1}, \ldots, y_N.$$

Conversely, suppose that

$$y_1, \ldots, y_{j-1}, y_L, y_R, y_{j+1}, \ldots, y_N$$

is an f-fringe, and let Γ be a fringed circuit in which $\langle y_i \rangle$ and $y_1, \ldots, y_{j-1}, y_L, y_R, y_{j+1}, \ldots, y_N$ are consecutive fringes.

Suppose that $p \in P_j$, and that V is the subset of variables in $x_1, x_2, \ldots, x_n$ which appear in p. Referencing a gate in Γ by the function of the inputs which it computes, a gate g is defined to be a p-gate if $p \leq g$. It then suffices to show that y_{j+1} or y_{j-1} is a p-gate.

Consider the digraph Π_Γ whose nodes are the p-gates of Γ, in which there is a directed edge from h to g if g is an input for h, and a bidirected edge if g and h are adjacent gates on some fringe of Γ.

Let y_r and y_s be p-gates on a fringe of Γ, and suppose that there are directed paths in Π_Γ from y_r and y_s to a common p-gate z. All gates y_t of the fringe, where $r \leq t \leq s$, are then also p-gates. To prove this, observe that a gate is a p-gate iff it evaluates to 1 under the assignment v to inputs in which precisely the inputs in V are 1. Since all gates on the connecting paths from y_r and y_s to z are 1 under the assignment v, so also are all intermediate gates y_t, as (by planarity) they are monotone boolean functions of the gates on the connecting paths.

Now let $W \subseteq V$ be the set of inputs which can be reached by tracing a directed path from y_j in Π_Γ. Certainly, W is non-empty, since at least one of the inputs to a p-gate is necessarily a p-gate. Let w be the assignment to inputs in which precisely the inputs in $V \backslash W$ are 1. Since f is 0 under w, and 1 under v, there is a gate on the fringe $\langle y_i \rangle$ which is also 0 under w and 1 under v. Any such gate y_t must be a p-gate, and moreover will be connected to an input in W by a path in Π_Γ. If t can be chosen distinct from j, then $t \leq j-1 \leq j$ or $j \leq j+1 \leq t$ and y_{j-1} or y_{j+1} is a p-gate. Otherwise, all gates other than y_j have the same evaluation under v and w. Since $y_1, \ldots, y_{j-1}, y_L, y_R, y_{j+1}, \ldots, y_N$ is an f-fringe, it follows that y_L or y_R has different evaluations under v and w, whence either y_{j-1} or y_{j+1} is 1 under v, and one or other is a p-gate.

■

As an immediate corollary to Theorem 2.1:

Cor.2.2:

If y_j is redundant in the f-fringe $\langle y_i \rangle$, then

$$P_j \subseteq P_{j-1} \cup P_{j+1} \text{ and } Q_j \subseteq Q_{j-1} \cup Q_{j+1}. \qquad (2.2)$$

Theorem 2.1 and Cor.2.2 show that if f is planar monotone computable from $\langle x_i \rangle$, then in any fringed circuit for f, each fringe is derived from its predecessor by performing a triop or by eliminating a redundant node subject to the necessary pre-conditions (2.1) and (2.2). In general, an operation on the fringe $\langle y_i \rangle$ (which may or may not be an f-fringe) is f-*admissible* if these necessary pre-conditions pertain. A sequence of fringes is then f-admissible if each fringe is derived from its predecessor by an admissible operation.

If $\langle y_i \rangle$ is a fringe and $p \in P_f$, then $y_r, y_{r+1}, \ldots, y_s$ is a p-*component* of $\langle y_i \rangle$ if $y_r, y_{r+1}, \ldots, y_s$ are p-gates, but neither y_{r-1} nor y_{s+1} is a p-gate. For $q \in Q_f$, the q-components of $\langle y_i \rangle$ are defined dually.

Suppose that $F_0, \ldots, F_k$ is an f-admissible sequence of fringes. Such a sequence can be viewed as a planar circuit Λ whose inputs are the functions in F_0, and whose outputs are those in F_k. As in the proof of Theorem 2.1, there is a digraph Π_Λ whose nodes are the p-gates in Λ, with a directed edge from g to h if h is an input gate for g, and a bidirected edge from g to h if g and h are adjacent on a fringe in Λ. The p-component A of F_0 is said to be *represented* by the p-component B of F_k if there is a directed path (possibly empty) in Π_Λ from a gate in A to a gate in B. The proof of Theorem 2.1 then shows that each p-component of F_0 is represented by exactly one p-component of F_k. Indeed, the fact that F_{i+1} is derived from F_i by an admissible operation is precisely the condition needed to guarantee that all the p- and q-components of F_i are represented on F_{i+1}.

The concepts of "p- and q-components of a fringe", and "representation of components of one fringe by components on another", are very helpful in analysing the geometry of fringed circuits. As explained above, if the fringe F_k is derived from F_0 by a sequence of f-admissible operations, each p- and q-component of F_0 is unambiguously represented in F_k. This makes it possible to refer to regions on a fringe through specifying components, and to correlate

regions on distinct fringes.

An algorithm for deciding planar monotone computability.

The algorithm for planar monotone computability developed in this section is based upon the geometrical relationship between p- and q-components. In essence, certain configurations of p- and q-components in a fringe are shown to persist throughout an f-admissible sequence of fringes, and a fringe is an f-fringe provided that it does not contain such a configuration either explicitly or in some latent form.

If y_u and y_v are p-gates on the fringe $\langle y_i \rangle$ such that no gate between y_u and y_v is a p-gate, then the p-components to which y_u and y_v belong are *adjacent* in $\langle y_i \rangle$. This pair of adjacent p-components is then *separated* by the q-component $y_r, \ldots, y_s$ if $u<r<s<v$.

A $\vee$*-pyramid* of degree n is a planar circuit with n inputs and $n(n+1)/2$ gates, in which each adjacent pair of inputs enters an $\vee$-gate, and the resulting n-1 $\vee$-gates are inputs to an $\vee$-pyramid of degree n-1. A $\wedge$-pyramid is defined dually.

Lemma 3.1:

Suppose that (for all q in Q_f) the adjacent p-components A and B of the fringe F_0 are separated by no q-component. There is an f-admissible sequence of fringes $F_0, \ldots, F_k$ such that A and B are represented by the same p-component in F_k.

Proof:

Consider the effect of building a truncated $\vee$-pyramid on $y_r, y_{r+1}, \ldots, y_s$; that is, a $\vee$-pyramid from which the last gate is removed. By Theorem 2.1, the construction of such a pyramid can be achieved by performing a sequence of triops, and leads to a fringe for f in which A and B are represented by the same component, provided that at no stage does an intermediate fringe develop a triple of consecutive gates x,y,z within the pyramid such that for some $q \in Q_f$:

$$q \geq y \text{ but } q \not\geq x \text{ and } q \not\geq z.$$

It is easy to see that the development of such a fringe is possible only if y represents a q-component separating A and B in F_0.

■

To understand the significance of being able to simplify the structure of the p- and q-components as described in Lemma 3.1, the functional implications of performing operations on fringes must be examined; when a fringe is altered by performing a triop or removing a redundant gate, the algebraic context of the computation is affected by the change of variables.

In effect, the following lemma describes the functional relationship between $f \equiv f(x_1, x_2, \ldots, x_n)$ and an arbitrary set of monotone boolean functions $y_1, y_2, \ldots, y_N$ in terms of the computational equivalence classes modulo f of the y_i's. It shows in particular that if $\langle y_i \rangle$ is an f-fringe the set of functions φ in FDL(n) such that $\varphi(y_1, y_2, \ldots, y_N) = f(x_1, x_2, \ldots, x_n)$ is an interval in FDL(N).

Lemma 3.2:

Let $f \in FDL(n)$. If $y_1,y_2, \ldots, y_N \in FDL(n)$, and $\varphi(e_1,e_2, \ldots, e_N) \in FDL(N)$, then

$\varphi(y_1,y_2, \ldots, y_N)=f(x_1,x_2, \ldots, x_n)$ iff $m(e_1,e_2, \ldots, e_N) \leq \varphi \leq M(e_1,e_2, \ldots, e_N)$

where

$$m(e_1,e_2, \ldots, e_N) \equiv \bigvee_{p \in P_f} \bigwedge \{e_i \mid y_i \geq p\} \text{ , and } M(e_1,e_2, \ldots, e_N) \equiv \bigwedge_{q \in Q_f} \bigvee \{e_i \mid y_i \leq q\}.$$

In particular: f is expressible as a function of $y_1,y_2, \ldots, y_N$ iff given (p,q) in $P_f \times Q_f$: there is a y_i such that $p \leq y_i \leq q$.

Proof:

By duality, it will suffice to show, given $p \in P_f$, that

$$\varphi(y_1,y_2, \ldots, y_N) \geq p \text{ iff } \varphi(e_1,e_2, \ldots, e_N) \geq \bigwedge \{e_i \mid y_i \geq p\}.$$

Now $\varphi(y_1,y_2, \ldots, y_N) \geq p$ iff $\varphi(y_1,y_2, \ldots, y_N)$ is 1 under the assignment mapping x_i to 1 if x_i appears in p and to 0 otherwise. Under this assignment, $y_i=1$ iff $y_i \geq p$, and thus an equivalent condition is: $\varphi(e_1,e_2, \ldots, e_N)=1$ under the assignment mapping e_i to 1 if $y_i \geq p$, and to 0 otherwise.

A necessary and sufficient condition for f to be expressible is $m \leq M$; this is the case iff each implicant of m has at least one variable in common with each clause of M.

∎

Lemmas 3.1 and 3.2 in conjunction indicate how a fringed circuit for f may sometimes be constructed by generating fringes in which the geometry of the p- and q-components becomes progressively simpler. For instance - observing that an f-admissible operation on a fringe cannot increase the number of p- or q-components - it may be possible to apply the technique of amalgamating components described in the proof of Lemma 3.1 repeatedly until (in some fringe $\langle y_i \rangle$) there is exactly one p-component for each p in P_f. By Lemma 3.2, it will then be sufficient to construct a planar circuit to realise the function $m(e_1,e_2, \ldots, e_N)$ as specified in Lemma 3.2 from the input sequence $e_1,e_2, \ldots, e_N$. An easy lemma shows this to be possible:

Lemma 3.3:

Every function of the form

$$g \equiv \bigvee_{t=1}^{m} g_t \text{ where } g_t \equiv \bigwedge_{k=i_t}^{j_t} y_k$$

and $i_1 < i_2 < \cdots i_m$ and $j_1 < j_2 < \cdots < j_m$ is planar monotone computable from $\langle y_i \rangle$.

Proof:

Each prime implicant g_t of g can be computed by a $\wedge$-pyramid; these pyramids can then be superimposed to obtain a planar circuit which computes the sequence of prime implicants $g_1,g_2, \ldots, g_m$ of g, and $\langle g_i \rangle$ is evidently an f-fringe.

Let **A** and **B** be sets of components of a fringe such that each component in **A** is a p-component for some p in P_f, and each component in **B** is a q-component for some q in Q_f. The configuration of components (**A**,**B**) is said to be *persistent* if

1. **A** and **B** contain a pair of adjacent components
2. **A** and **B** are closed under adjacency of components
3. any pair of adjacent components in **A** is separated by a component in **B**, and vice versa.

Note that (by condition 2) the sets A and B comprise all p-components and all q-components of the fringe, where p and q range over specified subsets of P_f and Q_f respectively. A persistent configuration in an f-fringe will remain persistent after the removal of a p-component from **A** if there is no longer a p-component in **A**; in this case, there can be only one p-component of the fringe, and this cannot separate an adjacent pair of q-components for any q in Q_f, since (by Lemma 3.2) there is some function y on the fringe such that $p \leq y \leq q$. Indeed, the same is true of any fringe whose constituent functions generate a sublattice of FDL(n) containing f. A simple extension of this argument shows that if such a fringe contains at least one pair of adjacent components, but no adjacent pair satisfying the premises of Lemma 3.1, then it contains a persistent configuration of components.

Lemma 3.4:

If (**A**,**B**) is a persistent configuration in F_0, and

$$F_0, \ldots, F_k$$

is an f-admissible sequence of fringes, then the representatives in F_k of the components in **A** and **B** satisfy the same relations of adjacency and separation relative to each other.

Proof:

It suffices to prove the lemma for k=1.

If A_1 and A_2 are adjacent p-components in F_0, then their representatives in F_1 are adjacent provided that they are distinct. Suppose then that A_1 and A_2 in F_0 have the same representative in F_1. There must be p-gates a_1 and a_2 in A_1 and A_2 separated on F_0 by a single gate g. There are two prima facie possibilities: F_1 is the result of performing a $\vee$-triop at g, or g is redundant, and F_1 is the result of removing g. By hypothesis, there is a q-component in **B** separating A_1 and A_2; thus g is a q-gate, but neither a_1 nor a_2 is a q-gate. By Theorem 2.1, neither of these operations on the fringe F_0 can then be f-admissible.

To ensure that separability relations are preserved, and so complete the proof, it suffices to prove that a p-component A in **A** and q-component B in **B** which are disjoint in F_0 have disjoint representatives in F_1. This follows by an easy case analysis from the fact that neither A nor B can have the same representative in F_1 as an adjacent component.

■

In combination, the above lemmas justify the following informal algorithm to determine whether a given monotone boolean function f can be computed by a planar circuit from the input sequence $x_1, x_2, \ldots, x_n$:

Algorithm 3.5:

begin
  F := $\langle x_i \rangle$
  while { the premise of Lemma 3.1 is satisfied for
      the pair of components (A,B) of the fringe F } **do**
    begin
      { construct an f-admissible sequence of fringes
          $F=F_0, \ldots, F_k$ (as in Lemma 3.1)
        so that A and B have the same representative in F_k }
      F := F_k
    end
  if { for every p in P_f, there is just one p-component
      in F, and dually }
  then { both F and $\langle x_i \rangle$ are f-fringes (Lemma 3.3) }
  else { F contains a persistent configuration, and neither F
      nor $\langle x_i \rangle$ is an f-fringe }
end

The correctness of Algorithm 3.5 depends upon Theorem 2.1, which shows that whether or not a fringe is an f-fringe is invariant under f-admissible triops - all operations on fringes performed in the **while**-loop are triops. Termination of the **while**-loop is guaranteed, as every iteration reduces the total number of components of the fringe F.

It may be seen that on termination either a planar circuit realising f from the input sequence $x_1, x_2, \ldots, x_n$ has been constructed, or a persistent configuration has been encountered.

A criterion for planar computability.

Algorithm 3.5 is useful as the basis for a practical technique for constructing a planar monotone circuit for a function, or proving that no such circuit exists. A more careful consideration of the amalgamation of components which occurs in the **while**-loop reveals a criterion for planar monotone computability which can be applied directly to $\langle x_i \rangle$ (Theorem 4.1). The relevant definitions and proofs are outlined here; the detailed proof of Theorem 4.1 is left to the reader.

Let $A_1, \ldots, A_k$ be a sequence of pair-wise adjacent p-components on a fringe maximal subject to the condition: for $1 \le i < k$, and for all q in Q_f, A_i and A_{i+1} are separated by no q-component. If y_r is the leftmost gate in A_1, and y_s is the rightmost gate in A_k, then $y_r, y_{r+1}, \ldots, y_s$ defines the p-*supercomponent* of the fringe spanned by $A_1, \ldots, A_k$. For $q \in Q_f$, the q-supercomponents of $\langle y_i \rangle$ are

defined dually.

Adjacency of supercomponents, and the concept of "a persistent configuration of supercomponents" can be defined in a manner precisely analogous to the corresponding definitions for components.

Theorem 4.1:

The monotone boolean function $f(x_1,x_2,\ldots,x_n)$ is computable from the left-right sequence of inputs $x_1,x_2,\ldots,x_n$ iff the input fringe $\langle x_i \rangle$ contains no persistent configuration of supercomponents.

Proof (sketch):

A fringe which contains a persistent configuration of supercomponents cannot be an f-fringe, since the analogue of Lemma 3.4 applies.

Conversely, if f cannot be computed by a planar circuit from the input sequence $x_1,x_2,\ldots,x_n$, then the **while**-loop of Algorithm 3.5 constructs a fringe $\langle y_i \rangle$ which contains a persistent configuration (**A**,**B**) of components. Since the fringe $\langle y_i \rangle$ is constructed from $\langle x_i \rangle$ by a sequence of admissible triops, it can be shown that for each component A in **A** (respectively B in **B**), the set of components of $\langle x_i \rangle$ represented by A (respectively B) in $\langle y_i \rangle$ spans a supercomponent of $\langle x_i \rangle$. Moreover, the system of supercomponents defined in this way is a persistent configuration in $\langle x_i \rangle$.

∎

As a corollary to Theorem 4.1, it may be seen that all monotone boolean functions of 4 or fewer inputs are planar monotone computable; a trivial analysis shows that no persistent configuration of supercomponents can be contained in the input fringe.

Concluding remarks.

A more explicit version of Algorithm 3.5, and a full analysis of its behaviour, will be the subject of a companion paper. The principles described in this paper have been used to develop a computer program (written by John Buckle) which can construct planar circuits for small boolean functions (e.g. T_2^7), and essentially decides planar computability.

Many aspects of this work merit further research. It would be of interest to know to what extent the arguments used here can be applied to the case when $x_1,x_2,\ldots,x_n$ are assumed to satisfy relations: the proof of Theorem 1.1 is one point at which freeness is required. It would also be helpful to develop upper bounds on the size of planar circuits constructed by Algorithm 3.5, and to devise variants of the algorithm which might improve these bounds.

The classification of "persistent configurations" is a possible approach to deriving an alternative characterisation of planar monotone computable functions. The simplest examples of such configurations are implicit in W.F.McColl's proof that T_3^5 has no planar realisation ([4] Theorem 1). In effect, McColl's proof shows that no function of 5 inputs a,b,c,d,e which has 'a∧c∧e' as a prime implicant, and whose prime clauses include either 'b∨e' or 'b∨d∨e' and

either 'a∨d' or 'a∨b∨d' is planar monotone computable from the input sequence <a,b,c,d,e>. As a second example, which illustrates that persistent configurations do not require an implicant or clause with more than 2 components, it may be seen that no function of 6 inputs a,b,c,d,e,f which has 'a∧d' and 'c∧f' as prime implicants, and 'a∨b∨c∨e' and 'b∨d∨e∨f' as prime clauses is planar monotone computable from the input sequence <a,b,c,d,e,f>.

In connection with persistent configurations, it is worth noting that: given a set P of conjunctions of literals, and a set Q of disjunctions of literals, the set of monotone functions $f(x_1,x_2,\ldots,x_n)$ such that $P \subseteq P_f$ and $Q \subseteq Q_f$ is a non-empty interval in FDL(n) provided that each conjunction in P has a literal in common with each disjunction in Q. Indeed, in lattice-theoretic terms, P and Q are sets of join- and meet-irreducibles respectively in FDL(n), with the property that $p \leq q$ whenever $p \in P$ and $q \in Q$. If $X \equiv \bigcup_{q \in Q} \{q_1 \mid q_1 \text{ is a meet-irreducible} < q\}$ and $Y \equiv \bigcap_{p \in P} \{p_1 \mid p_1 \text{ is a join-irreducible} > p\}$, then (using the notation of [1] §1) f lies in the interval:

$$m \vee \bigvee \tilde{X} \leq f \leq M \wedge \bigwedge \tilde{Y}.$$

Despite this simplification, the construction of persistent configurations still seems to present combinatorial problems.

Acknowledgements.

I am grateful to John Buckle for much programming assistance, and to Bill McColl for a number of helpful discussions and examples.

References.

[1] W.M.Beynon,
Replaceability and computational equivalence in finite distributive lattices,
Theory of Computation Report 61, University of Warwick, 1984.

[2] W.M.Beynon,
Replacement in monotone boolean networks: an algebraic perspective.
Proc. 4th FST&TCS, Bangalore, December 1984 (to appear).

[3] P.E.Dunne,
Some results on replacement rules in monotone boolean networks,
Theory of Computation Report 64, University of Warwick, 1984.

[4] W.F.McColl,
On the planar monotone computation of threshold functions,
Proc. 2nd STACS, Saarbrucken, January 1985 (to appear).